09.02.07 «Информационные системы и программирование»

Квалификация: программист

О Т Ч Е Т

|  |  |  |
| --- | --- | --- |
| по | производственной | практике |
|  | | |

|  |
| --- |
| Выполнил студент группы |
| Подъячев Владислав Алексеевич |
| *(ФИО)* |
|  |
| *(подпись)* |

|  |  |
| --- | --- |
| Проверили: |  |
|  | |
| *(должность, ФИО руководителя от* Выберите элемент.*)* | |
|  |  |
| *(оценка)* | *(подпись)* |
|  |  |
| МП | *(дата)* |
|  | |
| *(должность, ФИО руководителя от ГБПОУ МО «Физтех-колледж»)* | |
|  |  |
| *(оценка)* | *(подпись)* |
|  |  |
|  | *(дата)* |

***1. ТЕХНИЧЕСКАЯ РЕАЛИЗАЦИЯ СИСТЕМЫ ЭЛЕКТРОННЫХ МЕДИЦИНСКИХ КАРТ***

***1.1. Технологический стек***

*Для реализации системы электронных медицинских карт был выбран современный технологический стек, обеспечивающий высокую производительность, безопасность и масштабируемость решения.*

***База данных PostgreSQL 16***

*В качестве основной СУБД выбрана PostgreSQL версии 16, обладающая следующими преимуществами:*

* *Поддержка сложных типов данных и JSON для гибкой структуры медицинских записей*
* *Встроенные механизмы репликации и резервного копирования*
* *Полнотекстовый поиск с поддержкой русского языка*
* *Расширенные возможности по оптимизации запросов*
* *Соответствие требованиям ACID для критически важных медицинских данных*

***Язык программирования Python 3.11***

*Python выбран как основной язык разработки благодаря:*

* *Богатой экосистеме библиотек для работы с медицинскими данными*
* *Простоте интеграции с системами машинного обучения для анализа данных*
* *Высокой скорости разработки и поддержки кода*
* *Встроенным средствам для работы с криптографией*
* *Активному сообществу и регулярным обновлениям безопасности*

***Криптографическая защита***

*Для обеспечения конфиденциальности медицинских данных реализован комплексный подход к шифрованию:*

* *AES-256-GCM для шифрования диагнозов и персональных данных*
* *RSA-2048 для управления ключами шифрования*
* *SCRAM-SHA-256 для аутентификации пользователей*
* *TLS 1.2+ для защиты данных при передаче*

***Инструменты разработки и развертывания***

* *Docker для контейнеризации приложения*
* *Git для контроля версий*
* *PlantUML для документирования архитектуры*
* *Автоматизированные скрипты развертывания*

***1.2. Архитектура базы данных***

***Структура таблиц***

*Разработана нормализованная структура БД третьей нормальной формы (3НФ), включающая 5 основных таблиц:*

1. ***patients*** *- данные пациентов с зашифрованными персональными полями*
2. ***doctors*** *- информация о медицинском персонале*
3. ***appointments*** *- записи на прием с поддержкой различных статусов*
4. ***medical\_records*** *- медицинские карты с шифрованными диагнозами*
5. ***prescriptions*** *- назначения и рецепты*

***Оптимизация производительности***

*Для обеспечения быстрой работы системы создано 30 оптимизированных индексов:*

* *Составные индексы для частых запросов*
* *Полнотекстовые индексы для поиска по ФИО*
* *Частичные индексы для фильтрации по статусам*
* *Покрывающие индексы для отчетов*

***Обеспечение целостности данных***

* *Внешние ключи с каскадным удалением для связанных записей*
* *Проверочные ограничения (CHECK) для валидации данных*
* *Триггеры для автоматического обновления временных меток*
* *Уникальные ограничения для предотвращения дублирования*

***Резервное копирование***

*Реализована многоуровневая система резервного копирования:*

* *Полный backup ежедневно в 02:00*
* *Инкрементальные backup каждые 6 часов*
* *Архивирование WAL-логов непрерывно*
* *Автоматическая ротация старых резервных копий*

***1.3. Система безопасности***

***Шифрование данных (TDE)***

*Реализована система прозрачного шифрования данных (Transparent Data Encryption):*

* *Автоматическое шифрование чувствительных полей при записи*
* *Прозрачная расшифровка при чтении для авторизованных пользователей*
* *Ротация ключей шифрования каждые 90 дней*
* *Отдельные ключи для каждой таблицы с разным уровнем чувствительности*

***Аудит операций***

* *Логирование всех операций с медицинскими данными*
* *Сохранение истории изменений критических полей*
* *Невозможность удаления записей аудита*
* *Автоматические уведомления о подозрительной активности*

***Защита от атак***

* *Параметризованные запросы для защиты от SQL-инъекций*
* *Валидация и санитизация всех входных данных*
* *Ограничение частоты запросов (rate limiting)*
* *Автоматическая блокировка после неудачных попыток входа*

***1.4. Ключевые показатели производительности***

***Время отклика***

* *Поиск пациента: менее 5 секунд при базе в 1 миллион записей*
* *Создание новой записи: менее 2 секунд*
* *Генерация отчетов: менее 10 секунд*

***Пропускная способность***

* *Поддержка 1000+ одновременных пользователей*
* *Обработка до 10000 транзакций в час*
* *Масштабирование до 10 миллионов пациентов*

***Доступность системы***

* *Мониторинг 24/7 с автоматическими уведомлениями*
* *Автоматическое переключение на резервный сервер*
* *Целевой показатель доступности: 99.9%*

***2. РАЗВЕРТЫВАНИЕ И АДМИНИСТРИРОВАНИЕ***

***2.1. Стратегия развертывания***

***Контейнеризация с Docker***

*Приложение упаковано в Docker-контейнеры для обеспечения:*

* *Изоляции компонентов системы*
* *Простоты развертывания на различных платформах*
* *Воспроизводимости окружения*
* *Легкости масштабирования*

***Автоматизированная установка***

*Разработаны скрипты автоматической установки:*

* *Проверка системных требований*
* *Установка зависимостей*
* *Настройка базы данных*
* *Генерация ключей шифрования*
* *Создание начальных пользователей*

***Конфигурация через переменные среды***

*Все настройки системы вынесены в переменные окружения:*

* *Параметры подключения к БД*
* *Настройки безопасности*
* *Параметры производительности*
* *Расписание резервного копирования*

***Мониторинг состояния системы***

*Встроенная система мониторинга отслеживает:*

* *Загрузку CPU и памяти*
* *Использование дискового пространства*
* *Время отклика БД*
* *Количество активных сессий*
* *Ошибки и исключения*

***Автоматические обновления безопасности***

* *Регулярная проверка обновлений компонентов*
* *Автоматическая установка критических патчей*
* *Откат в случае неудачного обновления*
* *Уведомления администратора о проведенных обновлениях*

***Масштабирование по требованию***

*Система поддерживает горизонтальное масштабирование:*

* *Добавление новых узлов приложения*
* *Балансировка нагрузки между серверами*
* *Кеширование часто запрашиваемых данных*
* *Асинхронная обработка тяжелых операций*

***2.2. Репликация и отказоустойчивость***

***Streaming репликация PostgreSQL***

*Настроена потоковая репликация для обеспечения высокой доступности:*

* *Синхронная репликация на критических операциях*
* *Асинхронная репликация для отчетов*
* *Автоматическое обнаружение отставания реплик*
* *Мониторинг целостности репликации*

***Архитектура с 3 узлами***

*Реализована схема "1 мастер + 2 слейва":*

* *Мастер БД для операций записи*
* *Слейв 1 для операций чтения и горячего резерва*
* *Слейв 2 для аналитических запросов и backup*

***Балансировка нагрузки чтения***

* *Распределение запросов чтения между репликами*
* *Учет задержки репликации при маршрутизации*
* *Приоритизация по географической близости*
* *Автоматическое исключение отстающих реплик*

***Мониторинг задержки репликации***

*Система отслеживает:*

* *Лаг репликации в байтах и времени*
* *Скорость применения изменений*
* *Доступность всех узлов кластера*
* *Автоматические алерты при превышении порогов*

***Восстановление за 30 секунд***

*Процедура быстрого восстановления включает:*

* *Автоматическое обнаружение сбоя*
* *Выбор оптимальной точки восстановления*
* *Применение WAL-логов до актуального состояния*
* *Проверка целостности данных*

***2.3. Архитектура развертывания***

*Система развернута по схеме с тремя узлами, обеспечивающей высокую доступность и производительность:*

***Мастер БД (Основная запись PostgreSQL)***

* *Обработка всех операций записи*
* *Генерация WAL-логов для репликации*
* *Приоритетная обработка критических транзакций*

***Слейв 1 (Чтение/Резерв)***

* *Обработка операций чтения в реальном времени*
* *Горячий резерв для быстрого переключения*
* *Синхронная репликация с мастером*

***Слейв 2 (Чтение/Резерв)***

* *Обработка аналитических запросов*
* *Создание резервных копий без нагрузки на мастер*
* *Асинхронная репликация для производительности*

***2.4. Расписание резервного копирования***

***Полный backup ежедневно в 02:00***

* *Создание полной копии базы данных*
* *Шифрование backup-файла*
* *Проверка целостности после создания*
* *Отправка на удаленное хранилище*

***Инкрементальный backup каждые 6 часов***

* *Сохранение только измененных данных*
* *Минимальная нагрузка на систему*
* *Быстрое восстановление до любой точки*

***Архивирование WAL непрерывно***

* *Потоковое сохранение транзакционных логов*
* *Возможность восстановления до любой секунды*
* *Автоматическая компрессия старых логов*

***Тестирование восстановления еженедельно***

* *Автоматическая проверка возможности восстановления*
* *Восстановление на тестовый сервер*
* *Проверка целостности восстановленных данных*
* *Отчет о результатах тестирования*

***Заключение***

*Разработанная система электронных медицинских карт представляет собой современное, безопасное и масштабируемое решение для медицинских учреждений. Использование передовых технологий шифрования, репликации и резервного копирования обеспечивает надежную защиту критически важных медицинских данных при сохранении высокой производительности и доступности системы.*

*Конечно! Вот простыми словами теоретические ответы по каждому из пунктов, которые тебе могут пригодиться на производственной практике ПП.03:*

***1. Основы проектирования БД***

***• Этапы проектирования БД: концептуальный, логический, физический уровни.***

* ***Концептуальный уровень*** *— это разработка общей модели данных, без привязки к конкретной СУБД. Тут решают,* ***какие сущности*** *будут (например, "Пользователь", "Заказ"),* ***как они связаны*** *и* ***какие у них атрибуты****.*
* ***Логический уровень*** *— перевод модели в* ***таблицы****, поля, связи (например, первичные и внешние ключи). Учитываются* ***типы данных****.*
* ***Физический уровень*** *— конкретная реализация в выбранной СУБД:* ***индексы****,* ***табличные пространства****,* ***оптимизация хранения****.*

***• Принципы нормализации данных (1NF, 2NF, 3NF) и их применение.***

*Нормализация — это способ* ***упростить структуру БД и избежать дублирования****.*

* ***1NF (Первая нормальная форма):*** *в ячейке таблицы должно быть* ***одно значение****, а не список.*
* ***2NF:*** *таблица должна быть в 1NF, и* ***все неключевые поля*** *должны зависеть* ***от всего первичного ключа****, а не от части.*
* ***3NF:*** *таблица в 2NF и* ***отсутствуют транзитивные зависимости****, то есть, поля не зависят друг от друга, только от ключа.*

***• Отличия реляционной, иерархической и сетевой моделей данных.***

* ***Реляционная модель*** *— данные хранятся в* ***таблицах****. Таблицы связаны через ключи.*
* ***Иерархическая модель*** *— структура похожа на дерево:* ***один родитель, много потомков****.*
* ***Сетевая модель*** *— похожа на граф:* ***много связей*** *между записями, не только родитель–потомок.*

***2. SQL и СУБД***

***• Синтаксис DDL, DML, DCL команд в SQL.***

* ***DDL (Data Definition Language)*** *— управление структурой БД:*
  + *CREATE, ALTER, DROP*
* ***DML (Data Manipulation Language)*** *— работа с данными:*
  + *SELECT, INSERT, UPDATE, DELETE*
* ***DCL (Data Control Language)*** *— права доступа:*
  + *GRANT, REVOKE*

***• Особенности работы с транзакциями: ACID-свойства, уровни изоляции.***

* ***ACID*** *— требования к надежной транзакции:*
  + ***Atomicity (атомарность):*** *всё или ничего.*
  + ***Consistency (согласованность):*** *БД остается в правильном состоянии.*
  + ***Isolation (изоляция):*** *параллельные транзакции не мешают друг другу.*
  + ***Durability (долговечность):*** *после коммита данные не теряются.*
* ***Уровни изоляции:***
  + *READ UNCOMMITTED, READ COMMITTED, REPEATABLE READ, SERIALIZABLE*
  + *Чем выше уровень, тем меньше ошибок, но ниже производительность.*

***• Сравнение функционала СУБД (MySQL, PostgreSQL, MS SQL Server).***

* ***MySQL:*** *простой, быстрый, широко используется в вебе.*
* ***PostgreSQL:*** *мощный, поддерживает сложные запросы, расширяемый.*
* ***MS SQL Server:*** *хорошо интегрирован с Windows, удобные инструменты, платный.*

***3. Администрирование и безопасность***

***• Методы управления пользователями: роли, привилегии, аутентификация.***

* ***Роли*** *— группы прав, которые можно назначить пользователям.*
* ***Привилегии*** *— разрешения на действия: SELECT, INSERT, UPDATE, DELETE.*
* ***Аутентификация*** *— проверка личности пользователя (пароль, токен и т.д.).*

***• Алгоритмы резервного копирования и восстановления (полное/инкрементальное).***

* ***Полное копирование*** *— копируется вся база.*
* ***Инкрементальное*** *— копируются* ***только изменения*** *с последней копии.*
* *Обычно делают* ***полный бэкап раз в день****, а инкрементальные — чаще (например, каждый час).*

***• Защита от SQL-инъекций и методы шифрования данных.***

* ***SQL-инъекции*** *— атака, когда вредоносный код внедряется в SQL-запрос.*
  + ***Решение:*** *использовать* ***подготовленные выражения*** *(prepared statements), проверку ввода.*
* ***Шифрование:***
  + ***На уровне БД:*** *Transparent Data Encryption (TDE).*
  + ***На уровне приложения:*** *шифруем перед записью.*
  + *Используют* ***AES****,* ***RSA*** *и др.*

***Теоретические вопросы***

***1. Методы оптимизации запросов: индексы, материализованные представления***

* ***Индексы*** *— ускоряют поиск, как оглавление в книге.*
* ***Материализованные представления*** *— сохранённый результат запроса, обновляется по расписанию или вручную.*

***2. Архитектура клиент-серверных СУБД и особенности их администрирования***

* ***Клиент-сервер:*** *клиент отправляет запрос, сервер обрабатывает его.*
* *Особенности:*
  + *Настройка пользователей.*
  + *Мониторинг нагрузки.*
  + *Контроль доступа.*

***3. Технологии репликации и шардинга баз данных***

* ***Репликация*** *— копирование данных между серверами для отказоустойчивости.*
* ***Шардинг*** *— деление данных на части по ключу (например, по стране или ID) и хранение на разных серверах для масштабирования.*

***4. Механизмы аутентификации: Kerberos, OAuth 2.0 в контексте БД***

* ***Kerberos:*** *протокол с паролем и билетами, защищает от подделки личности, часто используется в корпоративных сетях.*
* ***OAuth 2.0:*** *чаще в веб-приложениях, пользователь даёт доступ без передачи пароля (например, вход через Google). Может использоваться при доступе к БД через API.*

***5. Принципы работы с журналом транзакций WAL (Write-Ahead Logging)***

* ***WAL*** *— журнал транзакций: сначала все изменения пишутся в лог, а потом — в БД.*
* *Это обеспечивает:*
  + *Восстановление после сбоя.*
  + *Поддержку атомарности и долговечности (из ACID).*

*Если хочешь — могу превратить это в шпаргалку, карточки для повторения или презентацию.*